Amendments to the Claims:

1-29 (Canceled).

30 (Currently amended): The method of claim ~~29~~ 58, wherein analyzing includes analyzing the vertex selected from a group consisting of a call vertex, an exit vertex, a conditional vertex, and a remainder vertex, and wherein the remainder vertex is a set that is the difference between the set of vertices and a set of call vertices, a set of exit vertices, and a set of conditional vertices.

31 (Original): The method of claim 30, wherein analyzing includes analyzing the vertex as a call vertex by transferring knowledge that an entry point of a called procedure is reachable and by transferring knowledge that the called procedure has been analyzed.

32 (Original): The method of claim 31, wherein transferring knowledge that an entry point of a called procedure is reachable includes executing the following acts: joining the set of path edges associated with the vertex with the transfer function associated with the vertex, self-looping the result of the act of joining, updating by providing the successor of the vertex as the vertex argument and the result of the act of self-looping as the path edge argument.

33 (Original): The method of claim 31, wherein transferring knowledge that the called procedure has been analyzed includes executing the following acts: joining the set of path edges associated with the vertex and the summary edges associated with the vertex, updating by providing the return point of the vertex as the vertex argument and the result of the act of joining as the path edge argument.

34 (Original): The method of claim 30, wherein analyzing includes analyzing the vertex as an exit vertex by analyzing an index in the set of successor indices of the vertex, wherein the

act of analyzing is iterated for another index until all indices in the set of successor indices are analyzed.

35 (Original): The method of claim 34, wherein analyzing each index includes executing the following acts: defining a call vertex as an element of a set of call vertices such that the index is equivalent to a return point for the call vertex, lifting a set of path edges associated with the vertex for a procedure containing the vertex, forming a union of the set of summary edges associated with the call vertex and a result of the act of lifting if the result of the act of lifting is not a subset of the set of summary edges associated with the call vertex, joining the set of path edges associated with the call vertex with the set of summary edges associated with the call vertex, and updating by providing the index as the vertex argument and a result of the act of joining as the path edge argument.

36 (Original): The method of claim 30, wherein analyzing includes analyzing the vertex as a conditional vertex by transferring knowledge to a true successor of the conditional vertex and transferring knowledge to a false successor of the conditional vertex.

37 (Original): The method of claim 36, wherein transferring knowledge to a true successor includes executing the following acts: joining the set of path edges associated with the vertex and a true transfer function associated with the vertex, and updating by providing the index of the true successor associated with the vertex as the vertex argument and a result of the act of joining as the path edge argument.

38 (Original): The method of claim 36, wherein transferring knowledge to a false successor includes executing the following acts: joining the set of path edges associated with the vertex and a false transfer function associated with the vertex, and updating by providing the index of the false successor associated with the vertex as the vertex argument and a result of the act of joining as the path edge argument.

39 (Original): The method of claim 30, wherein analyzing includes analyzing the vertex as a remainder vertex by executing the following acts: joining a set of path edges associated with the vertex and a transfer function associated with the vertex, obtaining an index from the set of successor indices of the vertex, updating by providing the index as the vertex argument and a result of the act of joining as the path edge argument, and iterating the act of obtaining to obtain another index and the act of updating until all indices in the set of successor indices are obtained.

40-42 (Canceled).

43 (Currently amended): The method of claim ~~42~~ 59, wherein finding includes choosing among multiple predecessor vertices that have the same length, wherein choosing includes choosing a predecessor vertex that produces a valuation of the vertex when a transfer function is applied to the predecessor vertex.

44 (Original): The method of claim 43, wherein finding includes finding a predecessor vertex that is a call vertex, wherein the transfer function, which includes a summary, is applied to the predecessor vertex.

45-47 (Canceled).

48 (Currently amended): The method of claim 60 ~~47~~, wherein finding the predecessor vertex such that two conditions exist, wherein one of the two conditions includes an existence of a path edge to the predecessor vertex in the set of path edges associated with the predecessor vertex at a ring one unit less than the ring of the reachable vertex.

49 (Original): The method of claim 48, wherein finding the predecessor vertex such that two conditions exist, wherein the other of the two conditions includes joining a path edge to the predecessor vertex with the transfer function at the predecessor vertex, and wherein the result of the act of joining contains a path edge to the reachable vertex.

50 (Canceled).

51 (Currently amended): The method of claim 61 ~~50~~, wherein finding the predecessor vertex such that two conditions exist, wherein one of the two conditions includes an existence of a path edge to the predecessor vertex in the set of path edges associated with the predecessor vertex at a ring one unit less than the ring of the reachable vertex.

52 (Original): The method of claim 51, wherein finding the predecessor vertex such that two conditions exist, wherein the other of the two conditions includes joining a path edge to the predecessor vertex with a set of summary edges associated with the predecessor vertex, and wherein the result of the act of joining contains a path edge to the reachable vertex.

53-55 (Canceled).

56 (Currently amended): The method of claim 63 ~~55~~, wherein finding a predecessor vertex according to the other of the two conditions, wherein the existence of the path edge to the predecessor vertex satisfies a transfer function at the predecessor vertex to form a successor vertex, and wherein the successor vertex is the reachable vertex.

57 (New): A computer-implemented method for checking a model of a program, comprising:

receiving a graph having a set of vertices and a successor function;

initializing sets of path edges, sets of summary edges, and a work list, wherein initializing includes setting each set of the sets of path edges to the empty set, wherein each set of the sets of path edges is associated with a vertex in the set of vertices, wherein initializing includes setting each set of the sets of summary edges to the empty set, wherein each set of the summary edges is associated with a vertex in a set of call vertices, wherein the set of call vertices is a subset of the set of vertices that represents call statements in the program;

removing a vertex having a type from the work list; and analyzing the vertex based on the type so as to determine the reachability status of the vertex in the set of vertices, wherein analyzing includes updating a set of path edges associated with the vertex by using a transfer function associated with the vertex.

58 (New): A computer-implemented method for checking a model of a program, comprising:

receiving a graph having a set of vertices and a successor function;

initializing sets of path edges, sets of summary edges, and a work list;

removing a vertex having a type from the work list; and

analyzing the vertex based on the type so as to determine the reachability status of the vertex in the set of vertices, wherein analyzing includes updating a set of path edges associated with the vertex by using a transfer function associated with the vertex, wherein updating includes executing the following acts: receiving a vertex argument and a path edge argument, forming a union of the set of path edges associated with the vertex argument and the path edge argument if the path edge argument is not a subset of the set of path edges associated with the vertex argument, and inserting the vertex argument into the work list.

59 (New): A computer-implemented method for generating a trace for a model of a program, comprising:

forming a control-flow graph having vertices from the model;

applying a transfer function to each vertex to form a set of path edges;

analyzing the set of path edges of a vertex;

tagging a unit length that the trace takes to reach the vertex from another vertex;

iterating the act of applying, analyzing, and tagging so as to form at least one trace to a vertex that is reachable in the model, wherein the at least one trace includes multiple unit lengths that form a length of the at least one trace; and

finding a shortest trace having a length, wherein the shortest trace is a subset of the at least one trace, wherein finding includes finding a predecessor vertex that has the length minus a unit length and iterating the act of finding the predecessor to find another predecessor vertex that has the length minus an additional unit length until no predecessor vertex can be found.

60 (New): A computer-implemented method for generating a trace for a model of a program, comprising:

forming a set of rings associated with each vertex of the model;

finding a ring such that a set of path edges of a reachable vertex exists; and

analyzing the reachable vertex based on a type of the reachable vertex so as to generate a trace from the entry of the main procedure of the program to the reachable vertex, wherein analyzing includes analyzing two cases if the reachable vertex is not an index of the first statement in a procedure containing the reachable vertex, wherein analyzing includes analyzing one of the two cases if a statement of the reachable vertex is not a skip statement immediately following a procedure call, wherein analyzing includes finding a predecessor vertex of the reachable vertex such that two conditions exist.

61 (New): A computer-implemented method for generating a trace for a model of a program, comprising:

forming a set of rings associated with each vertex of the model;

finding a ring such that a set of path edges of a reachable vertex exists; and

analyzing the reachable vertex based on a type of the reachable vertex so as to generate a trace from the entry of the main procedure of the program to the reachable vertex, wherein analyzing includes analyzing two cases if the reachable vertex is not an index of the first statement in a procedure containing the reachable vertex, wherein analyzing includes analyzing the other of the two cases if a statement of the reachable vertex is a skip statement immediately following a procedure call, wherein analyzing includes finding a predecessor vertex of the reachable vertex such that two conditions exist.

62 (New): A computer-implemented method for generating a trace for a model of a program, comprising:

forming a set of rings associated with each vertex of the model;

finding a ring such that a set of path edges of a reachable vertex exists; and

analyzing the reachable vertex based on a type of the reachable vertex so as to generate a trace from the entry of the main procedure of the program to the reachable vertex, wherein analyzing includes analyzing a predecessor vertex of the reachable vertex if the reachable vertex is an index of the first statement in the procedure containing the reachable vertex, wherein a statement associated with the predecessor vertex is a call to a procedure containing the reachable vertex, wherein analyzing includes finding the predecessor vertex and lifting a valuation associated with the reachable vertex to a path edge in the set of path edges associated with the predecessor vertex.

63 (New): A computer-implemented method for generating a trace for a model of a program, comprising:

forming a set of rings associated with each vertex of the model;

finding a ring such that a set of path edges of a reachable vertex exists; and

analyzing the reachable vertex based on a type of the reachable vertex so as to generate a trace from the entry of the main procedure of the program to the reachable vertex, wherein analyzing includes analyzing a predecessor vertex of the reachable vertex if the reachable vertex is an index of the first statement in the procedure containing the reachable vertex, wherein a statement associated with the predecessor vertex is a call to a procedure containing the reachable vertex, wherein analyzing includes finding a predecessor vertex according to two conditions, wherein one of the two conditions includes that the predecessor vertex be an element of a set of call vertices, and wherein the other of the two conditions includes an existence of a path edge to the predecessor vertex in the set of path edges associated with the predecessor vertex at a ring one unit less than the ring of the reachable vertex.